**What are the risks involved in building such a pipeline?**

* As [proof of concept (POC)](http://www.techopedia.com/definition/4066/proof-of-concept-poc/) is a demonstration aimed at verifying that certain concepts or theories can be achieved. A prototype is designed to determine feasibility but does not represent the final deliverable.
* Considering the scope of the POC and the size of the data,
  + Various steps of data preprocessing like data cleaning, validation is not considered.
  + It is usually delivered quickly and without extensive testing so thorough testing needed on the result set.
* With all above mentioned things, variation of values in the feature set of the dataset resulted with the presented data pipeline may not yield the same results with large data set.
* Design by prototyping does not fulfill all the requirements. There are a lot of important things like to specify performance criteria, storage constraints, security details and critical non-functional requirements.
* Assumptions on the columns that make tweets unique to avoid duplicates and to count unique tweets.
* Assumptions on tweets related to music just based on variations of word music as there could be other synonyms and context related to music without explicitly mentioning ‘music’
* As every database has its own advantages and disadvantages, a straightforward approach for storing the above POC data would be to use an RDBMS like MySQL since we require joins. But relational databases come with their challenges, especially when we need to **scale** them.
* One of the challenges could be to maintain the Twitter API connectivity on developer account. Sometimes elevation of account is needed to access extra features.
* Analysis should consider the percentage of people using twitter and restrictions on usage of twitter worldwide.
* **Security threat:** Collection of social media public data might be vulnerable to threats with virus files which may attack the company system resources and may cause concerns with data breach.
* Social media threats with data collection like rights to use the Justin Bieber tweet data and privacy levels with twitter data if used for bad promotions and causing reputation issues.

Having a better plan to build the POC that simulates the real time production level data pipeline involving design of the model, data analysis, ingestion, cleaning, transformation, loading and testing not only the data across the data pipeline but also the result set data helps mitigate the risks involved in building the data pipeline even at the POC level.

**How would you roll out the pipeline going from proof-of-concept to a production-ready solution?**

**Proper planning:**

Things don’t always go the way we think. We will face many problems during and after the POC to production phase.

So, there is need to have a better planning from gathering requirements to build the pipeline that yields machine learning ready dataset which would help the business make decisions.

Initially thinking through and good documentation of the following stages of data pipeline help us dig into the specifications at a high level,

* Business Understanding
* Project Plan
* Initial Data Collection
* Data Description
* Data Selection
* Data Cleaning
* Data Derivation
* Data modeling

Additionally, setting a **tangible timeline** for each phase of development of data pipeline is important.

**Proper Design and Architecture:**

Its vital to have a clear understanding of the entire data flow by **designing and clarifying** things to build best data pipeline that serves the **historical** and **incremental** data load easily

In case of our data model, keeping the huge data size in mind is important.

**Tools and Infrastructure:**

Based on all the requirements and scope of the project, tools and resources have to decided.

Like **ETL tools** to handle real time Twitter Streaming data, best Orchestration tools

For Example, Deploy Operators and DAGs to AWS hosted **Apache Airflow** and execute your Data Pipelines with DAG and Data Lineage Visualization.

**Cloud services** for file storage like s3 (store photos and videos), Spark EMR Cluster

**Storage Options:**

SQL vs NoSQL

Though SQL maintains better relationships and ACID properties, Production level data needs to be handled with in large scale and with file based storage in Twitter data, NoSQL has to be used.

* We can store photos in a distributed file storage like [HDFS](https://en.wikipedia.org/wiki/Apache_Hadoop) or [S3](https://en.wikipedia.org/wiki/Amazon_S3)
* Additionally, we can use a wide-column datastore like [Cassandra](https://en.wikipedia.org/wiki/Apache_Cassandra) to store relations like ‘UserPhoto’ and ‘UserFollow’.

**Incremental data load:**

As the number of tweets per second grows heavily, Scheduling the incremental data load with the real time Twitter data needs to planned and scheduled appropriately

Necessary code changes data transformations to deal with the incremental (Change Data Capture CDC data) like having a timestamp columns.

**Testing:**

With data quality testing, we can be reasonably confident of the quality of our data. But there can be unexpected skews in data size, skews in the number of rows dropped, etc. In order to catch unexpected changes in data size, we need to monitor our data pipeline.

Testing Phases like local, development and production help too.

**Non-functional Requirements (Extra features):**

Scalability (Not just Justin Bieber but for other keywords search)

Building pipeline that would help extract features for multiple ML models

Availability of the data (datalake) in case of resource failures by Replication

**What would a production-ready solution entail that a POC wouldn't?**

Production ready solution involve a lot of things and considerations which POC would not involve.

* In addition to the things mentioned like **Proper Planning, documentation, design and architecture, choosing tools and infrastructure, better storage options, Incremental data load, Testing** in the above solutionthere would be further special considerations as we dive deep into each phase
* Dealing with big data (Twitter Streaming Justin Bieber Data) in all the phases needs thorough **exploratory data analysis** especially before data ingestion i.e., even before pulling tweets from Twitter API. Enough testing is needed on the pulled data helps build better pipeline.
* As Twitter data is unstructured (media content), **data cleaning and validation** is very much essential to carry the data smooth to further data flow.
* **Data Duplication** are inevitable in big data, handling them is a must. So, filtering the data or views with our own logic to extract and transform data could avoid data redundancy.
* **Encryption** of data to maintain data secure will be needed for private user tweet data.
* **Incremental data** retention period for incremental data to resolve issues if any production issues or threats.
* **End Results with production level data would yield better and accurate analysis** where assmall data in POC with rare occurences would not.
* **Constant Twitter API connectivity** on developer account and sometimes elevation of account is needed to access features entertained. It would be hard to rerun the pipeline in case of API connection loss.
* **Better Computational resources** for performance enables optimal run of the data pipeline at a larger scale.
* Storage of the end machine learning ready dataset and testing it at various phases is a mandatory before confirming the correctness of ML ready dataset.

To conclude, there are various aspects of real time data which has to be kept in mind to deal with production level data pipelines as it would be **tough to roll back** any of the phases of data pipeline.

**What is the level of effort required to deliver each phase of the solution?**

As production level solution deals with big data, lot of efforts are invested to build data pipeline.

Some of them are highlighted below,

Proper **Business Understanding** of the required dataset for ML models helps build better data flow projects.

Good **estimates on the timeline** and usage of resources reduces efforts in the phases.

Data Ingestion (**Schema compatibility**)

Extracting Twitter Streaming Justin Bieber Data and filtering out columns should match the required schema for the model. More steps can be taken to evaluate the schema check to ensure correctness data columns pull from the API source.

Testing and ensuring data good as per the requirements from the first phase of pipeline reminds the saying ‘***Well Begun is Half Done’***

Handling constant Twitter API connection and having **a backup plan** for any issues.

**Data Pipeline scheduling and monitoring** for both historical and more necessarily incremental data such as Manual effort to monitor and automation to trigger the time line based pipelines.

Data Cleaning:

**Data discrepancies with** special characters, null values replacement, trimming data if necessary and also handling special symbols(encode and decode if needed) like emojis needs to handled as unclean data is like garbage to the end ML model.

**Storage** of the unstructured data like photos videos anticipating future ML model usage.

Data Transformation:

Tokenizing the tweet data to find music related tweets and also filtering of data to capture music related tweets. Dealing with streaming data to all the data transformations needed enough computational resources like EMR Cluster to run spark jobs.

**Uniqueness** of the data retrieved to extract tweetCount. Analysis is needed to judge a tweet to be unique like tweet id.

**Testing** the data at **phases** is always a need to avoid the incorrect data flow.

Efforts to **maintain** and **store** the end result data set to deliver to ML models.

Additionally, there would be lot of efforts needed to maintain, monitor, adjust the resources and services that helped to build the entire pipeline. Parallel run of the resources to work with big data would reduce some of the efforts.

**What is your estimated timeline for delivery for a production-ready solution?**

As the answer to this is based on various factors, assuming current twitter data size and requirement use case, considering 3-5 people on team:

I would think the range from minimum of 1 month to 3 months for overall development and testing for delivery for a production-ready solution